CONTEXT-SENSITIVE KEY MANAGEMENT FOR SMART GRID TELEMETRIC DEVICES

BY

REHANA TABASSUM

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Computer Science
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2013

Urbana, Illinois

Adviser:

Professor Klara Nahrstedt

ABSTRACT

In smart grid, the scale of pole devices that monitor the health of power line is large. With the upgrade of smart grid, the number of these resource-constrained (in terms of memory and computation) devices is further increasing. These devices are easy targets to security attacks as they are accessible via wireless network, and use weak passwords for authentication and transferring telemetric data to the pole maintenance personnel. General-purpose security protocols are not suitable for providing data security to these devices with limited memory, computational power and network connectivity. Therefore, security in smart grid is still a challenge.

In the first part of this thesis, we present a SCalable and Automated PAssword-CHanging protocol, SCAPACH, for unique authentication of human personnel (operator) and secure collection of telemetric data from a large number of measurement devices. SCAPACH employs physical per-operator, per-pole-device information as well as changeable secret salts to generate new unique passwords and secret keys every time a pole device is accessed. In this work, we address the memory and computational constraint problem of measurement devices. Besides, we address the limited change management capability problem of the measurement devices and our protocol works for evolving infrastructure. Our experiments confirm that the password-changing protocol authenticates and transmits measurement device data securely and in real-time under varying maintenance scenarios.

In the second part of this thesis, we describe a secure and lightweight scalable security protocol that allows a power system operator to collect data from measurement devices using data collectors. The security protocol trades off between computations and device memory requirements and provides flexible association between data collectors and measurement devices. These features allow data to be securely transferred from measurement devices to power operator via mobile or untrustworthy data collectors. We analyze the complexity and security of the protocol and validate its performance using experiments. Our results confirm that the protocol collects data in a secure, fast and efficient manner.

To my family, for their love and support.

ACKNOWLEDGMENTS

First and foremost, I would like to thank my advisor, Professor Klara Nahrstedt, without whose constant guidance and support, this work would not have been possible. She has been an extremely motivating and helpful mentor, and has made the journey towards completion of the thesis smoother. I would like to thank Edmond Rodgers, who has first came up with the problem, for helping me with his vast industrial experience. I also thank Gyorgy Dan and King-Shan Lui for their thoughtful suggestions about my work and teaching me different aspects of smart grid and security. Both of them actively helped me in my dissertation research.

I am grateful to the members of the MONET group, who have always provided me an atmosphere that is both stimulating and enjoyable. I also want to thank all the members of TCIPG for their support and valuable suggestions on my work. This material is based upon work supported by the Department of Energy under Award Number DE-OE0000097.

I thank my Bangladesh Community in Urbana-Champaign for their company and love; without them it could have been very hard to live being away from my family.

I am thankful to my wonderful family. I cannot express sufficient gratitude to my parents - my father M A Samad, my mother Jahanara Samad and my elder brother M A Ahad. Their unconditional love and blessings have enabled me to come this far in life. Last but not the least, I thank my wonderful husband Ahsan to help me in every aspect of my life. He has always been an extreme motivation of my work.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Motivation

Current power grid systems and their power lines in the field are monitored by telemetric measurement devices, which are sensors with capacitor banks. These devices are placed on top of electric poles. These devices usually measure telemetric measurements like frequency, voltage, and current readings from power lines and store them locally. Utility companies collect data readings from these measurement devices on a regular basis to ensure that the health of power line is sound and stable. The proper operation of the power grid relies on the quality of data collected from this large number of sensors and measurement devices. The data collection needs to be efficient and secure in order for smart grids to be economical and dependable. For example, these data are critical when damages occur due to any disaster. The utility company needs to identify the faulty location by frequently analyzing the unusual data readings taken from these measurement devices.

In our hierarchical data collection model, human operator carrying data collector devices are responsible for collecting data from the measurement devices. The data collector devices report the data readings back to the utility company to ensure better *situational awareness*; where *situational awareness* is the ability to know what is happening on the grid and to anticipate future problems in order to take effective actions [36].

The motivation of the thesis includes:

> **Scalability:** Power grid system uses a large number of measurement devices. On the other hand, the number of data collector devices may vary from small to large number according to the requirement of a particular utility company. Therefore, the data collection framework should scale well with the changeable architecture and vast number of measurement and data collector devices.

Flexibility: The data collection framework should provide flexible association between measurement devices and data collector devices. In other words, the utility should have the flexibility to assign different data collector devices to collect data from the same measurement device at different times. This feature is particularly important in scenarios where the data collector devices are mobile or the infrastructure is evolving so that different mappings or assignments between the data collector devices and measurement devices can be used at different times after the deployment of the measurement devices.

Security: Currently security of data inside the measurement devices and the data collector devices is an important concern. Data inside the measurement devices and data collector devices are easy target to security attacks due to the wireless channel over which data is transmitted, and also due to the weak passwords and vulnerable authentication protocol that utilities use to access theses devices. The security threats are further increasing with the increased scale of these small resource-constrained devices due to continual security reviews and cryptanalysis advancements [2]. Measurement devices are typically secured by simple passwords, known to many users (maintenance personnel or operators), with the same password often used for a large number of devices. Besides, telemetric measurements are transmitted over wireless channel by encrypting with the same symmetric key stored in both devices in every communication. The password used by human operator and the keys used for encryption should be changed periodically.

The framework should maintain the confidentiality and integrity of the data readings. In other words, the data reported by the measurement devices to the utility should remain secret to a potential eavesdropper and an active attacker. In addition, some systems require the data collector device to perform an integrity check right after it receives the message containing the data from the measurement device, even though it may not decrypt the message to retrieve the raw data. This enables a trustworthy data collector to immediately detect potential data corruption and/or tampering, so that a remedial action can be taken. Without this feature, corrupted and/or tampered data cannot be detected until delivered to the utility. The utility should also be able to tell whether the data have been tampered with. In some systems it is also required that the data to remain confidential to a compromised data collector device. This allows the utility company to outsource the data collection procedure without sacrificing

security. In the event that the data collector needs to aggregate the data, *homomorphic encryption* can be used to maintain data confidentiality.

Resource-limitation: The measurement devices limited storage and computational capacity. Therefore, the data collection framework needs to be computationally lightweight for the measurement devices, i.e., the framework should require the measurement devices to perform very few expensive cryptographic operations and to send few messages over wireless network.

Therefore, the development of a robust, scalable password-changing and data collection protocol framework is imperative to ensure secure device authentication and secure delivery of data within real-world constraints.

1.2 Problem Description and Contribution

In the first part of the thesis we focus on the problem of (i) authentication of human operator and data collector device and (ii) data collection by data collector device from measurement devices. In the first part we assume that the data collector devices are trustworthy and are allowed to read the data reported by the measurement devices. In addition, we assume that data collector devices report data to the control center when they are within the control center's security perimeter.

We propose a fast, cost-effective, scalable, and robust password-changing protocol framework, SCAPACH, which generates new device passwords to be used for **authentication between data collector and measurement devices**, and symmetric keys to be used for **secure data communication**. To the best of our knowledge, this is the very first attempt to address our goals. We introduce *Physical Unclonable Functions (PUFs)* to alleviate the load of measurement devices in generating and keeping keys without revealing them. It lessens the memory and computational burden from measurement devices. Moreover, our SCAPACH protocol generates device passwords and symmetric keys based on physical information (such as local time, pole geographical location, data collector device id etc.) and changeable stored secret. Hence, they are short-lived. We ensure that -

i. different device passwords and symmetric keys are generated inexpensively, and used every time an operator accesses a measurement device using his/her data collector device and

ii. data are transmitted in a secure and real-time manner.

To validate its performance we implemented the SCAPACH protocol. Our experiments confirm that our password-changing protocol authenticates and transmits pole device data securely and in real-time under varying maintenance scenarios.

Although we assumed that the data collector devices belong to the control center, there can be situations where the data collector devices are considered untrustworthy. Besides, extending the security perimeter to a vast number of data collector devices might be expensive and can even be infeasible when the data collection architecture is changeable or data collector devices are mobile. In these cases, the data collector devices do not need to read or understand the data but only relay the data back to the control center. So, the measurement devices need to encrypt the data in a way so that the data collector device does not have access to data even if it carries them. In addition, it needs to ensure that the data will remain secure even if the data collector device is compromised. To solve this problem, other researchers, Gyorgy Dan and King-Shan Lui, extended our SCAPACH approach and designed a key establishment and data collection protocol in [1]. The protocol named SELINDA allows data to be securely transferred from measurement devices to power system operator via mobile or untrustworthy data collectors. Moreover, the focus was to design the protocol as computationally lightweight as possible (i.e., minimizing the number of expensive cryptographic operations and the number of messages exchanged between devices). I implemented the protocol, ran several experiments to see the performance, and we jointly reviewed the design of the protocol. In the second part of our thesis, we describe the SELINDA protocol along with the experimental results presented in [1]. Our experimental results confirm that SELINDA protocol collects data in a secure, fast and efficient manner.

1.3 Thesis Outline

We first present some background information about cryptographic algorithms in chapter 2. In chapter 3 we look at some of the related works. In chapter 4 we define the model and architecture considered. We present the SCAPACH framework for secure authentication and delivery of data in chapter 5. We describe the SELINDA framework for secure delivery of data by measurement devices to the power operator using mobile and untrustworthy data collector device in chapter 6. We conclude in chapter 7 with the discussion along with the future work.

CHAPTER 2

BACKGROUND

In this chapter we discuss some background schemes that are used later in the thesis. We describe symmetric and asymmetric key cryptography, cryptographic hash function and physical Unclonable function.

2.1 Symmetric Key Cryptography

Symmetric key cryptography is an encryption system in which the sender and receiver of a message share a single, common key that is used both to encrypt and decrypt the message. In practice the keys represent a shared secret between two or more parties that can be used to maintain a private link for communication. Symmetric-key cryptography is also known as secret-key cryptography and private key cryptography. Figure 2.1 describes the symmetric key cryptography.

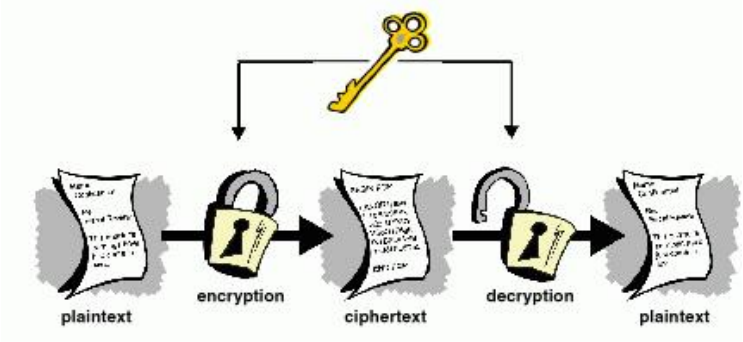

Figure 2.1: Symmetric Key Cryptography

Symmetric-key cryptography is simpler and faster, but its main drawback is that the two parties must somehow exchange the key in a secure way. Widely used symmetric key algorithms are Advanced Encryption Standard (AES), RC4, Blowfish, Data Encryption Standard (DES), and triple DES. In our

experiments AES is used as symmetric key algorithm to encrypt telemetric readings. High speed and low RAM requirements were criteria while designing the AES process. Thus AES performs well on a wide variety of hardware, from 8-bit smart cards to high-performance computers. Below is a short description on how AES works.

Advanced Encryption Standard (AES): AES [34] is iterative, symmetric-key block cipher that can use keys of sizes 128, 192, and 256 bits. It encrypts and decrypts data in blocks of 128 bits (16 bytes). The input block of 128 bits is arranged in the form of a matrix of 4 × 4 bytes (Figure 2.2). AES uses a loop structure that repeatedly performs permutations and substitutions of the input data.

$$\begin{bmatrix} byte_0 & byte_4 & byte_8 & byte_{12} \\ byte_1 & byte_5 & byte_9 & byte_{13} \\ byte_2 & byte_6 & byte_{10} & byte_{14} \\ byte_3 & byte_7 & byte_{11} & byte_{15} \end{bmatrix}$$

Figure 2.2: Representation of Input/Output Blocks in AES

The overall structure of AES encryption/decryption for 128-bit key size is shown in Figure 2.3. IN AES, the encryption key is first expanded into a key schedule that consists of 44 4-byte words. Encryption includes 10 rounds of processing for 128-bit keys, 12 rounds for 192-bit keys, and 14 rounds for 256-bit keys. Except for the last round in each case, all other rounds are identical. Each round of processing includes one single-byte based substitution step, a row-wise permutation step, a column-wise mixing step, and the addition of the round key step. The order in which these four steps are executed is different for encryption and decryption (Figure 2.3).

For encryption, the first three steps are permutation and substitution on input block data, whereas, the last step consists of XORing the output of the previous three steps with four words from the key schedule. These four rounds are called SubBytes, ShiftRows, MixColumns, and AddRoundKey. During SubBytes, a lookup table is used to determine what each byte is replaced with. The ShiftRows step has a certain number of rows where each row of the state is shifted cyclically by a particular offset, while leaving the first row unchanged. Each byte of the second row is shifted to the left, by an offset of one,

each byte in the third row by an offset of two, and the fourth row by an offset of three. The MixColumns step is a mixing operation using an invertible linear transformation in order to combine the four bytes in each column. The four bytes are taken as input and generated as output. In the fourth round, the AddRoundKey derives round keys from key schedule, and adds the round key to each byte of the state. Each round key gets added by combining each byte of the state with the corresponding byte from the round key.

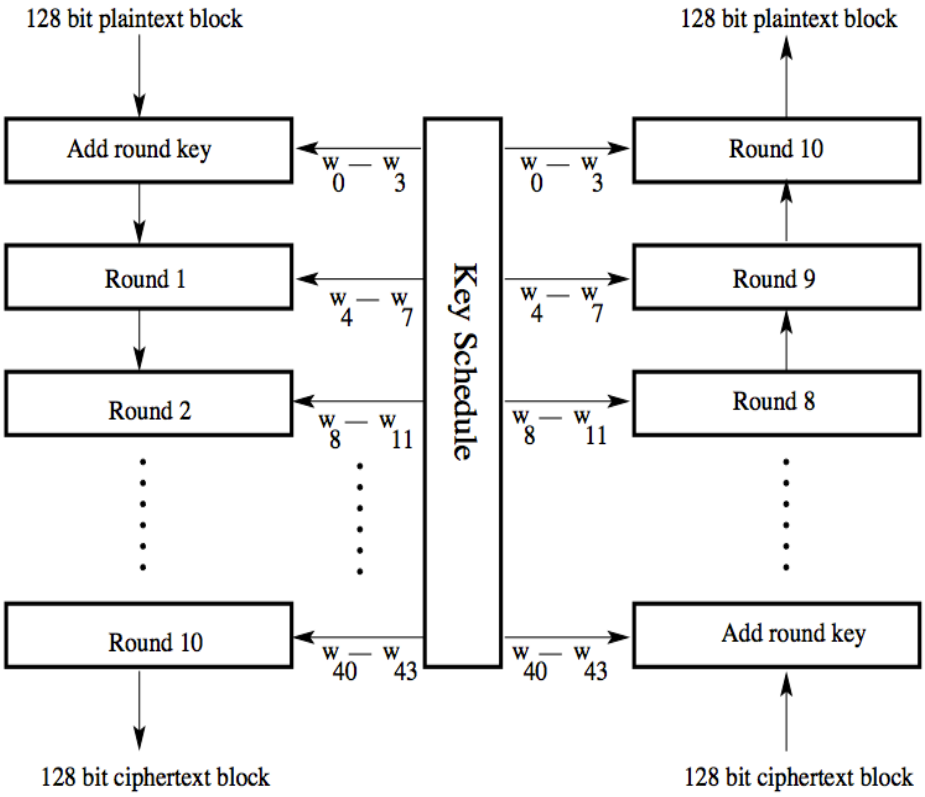

Figure 2.3: AES Encryption/Decryption for 128-bit Key

For decryption, each round consists of the following four steps: inverse shift rows, inverse substitute bytes, add round key, and inverse mix columns. The inverse steps perform the opposite transformations of each corresponding steps. The third step consists of XORing the output of the previous two steps with four words from the key schedule.

2.2 Asymmetric Key Cryptography

An asymmetric key cryptography is a cryptographic system that uses a key pair that includes a public key, known to everyone, and a private or secret key, known only to the recipient of the message. The two parts of this key pair are mathematically linked. The public key is used to encrypt plaintext or to verify a digital signature; whereas the private key is used to decrypt cipher text or to create a digital signature.

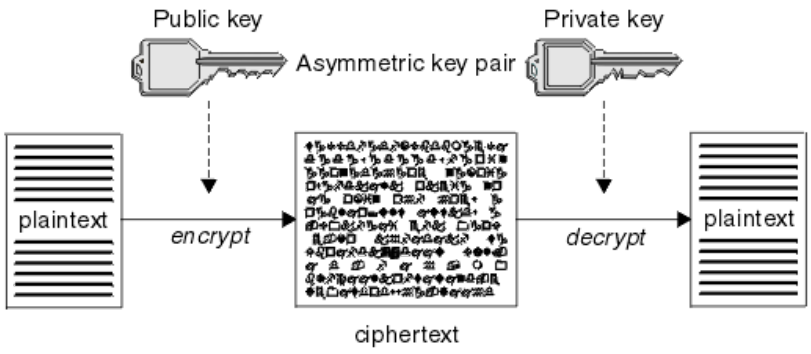

Figure 2.4: Asymmetric Key Cryptography

Therefore two main uses for public-key cryptography are:

i. Public-key encryption, in which a data is encrypted with the receiver's public key. The message cannot be decrypted by anyone who does not have the matching private key. Thus he presumed to be the owner of that key and the person associated with the public key. This ensures the confidentiality of the message.

ii. Digital signatures, in which a message is signed with the sender's private key and can be verified by anyone who has access to the associated public key. This verification proves that the sender had access to the private key, and therefore is the person associated with the public key. This also ensures that the message has not been tampered with, since any manipulation of the message will result in changes to the encoded message digest. This ensures the integrity of the message.

Widely used asymmetric key algorithms are Diffie–Hellman key exchange protocol, RSA, DSS (Digital Signature Standard), ElGamal etc. Following is the description of how Diffie-Hellman key exchange works.

Diffie-Hellman key exchange (DH): The Diffie-Hellman key exchange [32] works as follows: when Alice and Bob want to establish a shared key, Alice picks a natural number *a* and keeps as secret. Similarly, Bob picks a natural number b as his secret. Alice sends $g^a \bmod p$ to Bob, and Bob sends $g^b \bmod p$ to Alice. When Alice receives $g^b \bmod p$, she computes the shared key by $(g^b \bmod p)^a \bmod p = g^{ab} \bmod p$. Bob, on the other hand, computes the shared key by $(g^a \bmod p)^b \bmod p = g^{ab} \bmod p$. Note that although an eavesdropper knows p, g, $g^a \bmod p$, and $g^b \bmod p$, it is very difficult for him to compute $g^{ab} \bmod p$. In this thesis, we call $g^a \bmod p$ and $g^b \bmod p$ DH half keys or DH public keys.

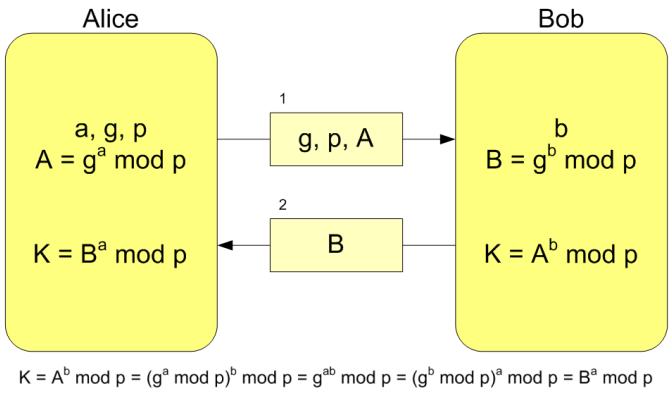

Figure 2.5: Diffie-Hellman Key Exchange Algorithm

2.3 Cryptographic Hash Function

A cryptographic hash algorithm is designed to provide a random mapping from a string of binary data to a fixed-size hash value, also called “message digest”, and achieve certain security properties. Hash algorithms can be used for digital signatures, message authentication codes, key derivation functions, pseudo random functions, and other security applications in the information infrastructure. [35]

In this thesis, we use cryptographic hash function to generate new key or password. Well-known hash functions are MD4, MD5, SHA-1 and SHA-2, SHA-3. We use SHA-2 in this thesis as hash function to generate new key or password.

2.4 Physical Unclonable Function

A PUF implements an on-chip physical function puf: C → R that takes an input challenge $Ch_i \in C$ and produces a response $Rs_i \in R$, where (C, R) is the set of all possible challenge-response pairs (CRPs). PUF relies on the intrinsic randomness during the integrated circuit fabrication process [14]. Therefore, CRPs cannot be cloned or reproduced exactly, not even by its original manufacturer, and is unique to each PUF [15].

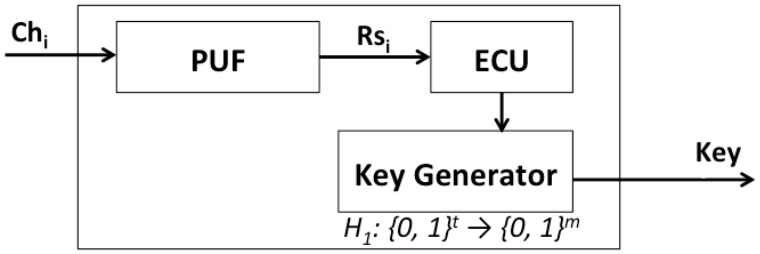

Figure 2.6: Key Generation Using PUF

PUF can generate volatile cryptographic keys with low-cost [3] when a challenge is given. In practice, error correction codes (e.g., Reed-Solomon) are used to remove the noise from the PUF response and make it stable and identical. The output of the error correction unit (ECU) of length t is hashed down by the hash function $H1: \{0, 1\}_t \rightarrow \{0, 1\}_m$ to a desired key KC of length *m*. KC is used for communication between DC and MD during the initial setup (detail in Section III). The key generation process using PUF is shown in Figure 2.6.

CHAPTER 3

RELATED WORK

Security in smart grids is a challenging problem for many reasons [22], [23]. One of the biggest challenges comes from connecting the grid with a plethora of devices with limited memory, computational power and network connectivity [22]. Furthermore, interoperability and legacy-compliance are also key concerns [23]. Therefore, general-purpose Internet security protocols are not adequate, and new security protocols tailored for the smart grid need to be developed.

Password verification problem over an insecure network has been investigated for a long time. Many existing security solutions have been built based on Diffie-Hellman (DH) key exchange protocol [32]. In 1992, Bellovin and Merritt [33] proposed Encrypted Key Exchange (EKE) protocol, which is a password-authenticated key agreement method, based on RSA [31] and DH [32]. Protocols such as SPEKE [8], DHEKE [4], A-EKE [6], and SRP [5] have been proposed in later time, which are strongly secured protocols of the EKE family. These approaches are computationally expensive. Besides, the number of messages exchanged between the two parties is not trivial. Since the Measurement devices have limited storage and computational capacity, lightweight (in terms of memory and computation overhead) protocols for measurement device is needed to perform associated cryptography [13]. Thus existing general-purpose protocols are not suitable for smart-grid systems.

Secure authentication for smart grids has been considered in [24], [25]. The focus of these mechanisms is on how to establish a shared key for data authentication between two entities. The two entities need to establish a session, which may be infeasible in the hierarchical data collection model. [21] studies how to access power system devices remotely for substation monitoring. The substation controller (or data concentrator) is used as the central point of access control. The substation controller authenticates users and keeps access logs, and is assumed to be trusted. However, the issue of securing data between the devices and the utility is not considered in this work.

Key management is another important issue in security. Traditional PKI systems (e.g., X.509) are not used in smart grid due to their structural complexity and cost for establishing and managing the framework. As compared to the PKI systems, Identity-Based Cryptography (IBC) is much simpler [11].

Shamir [11] introduced the concept of IBC and since then many ID-based key agreement protocols have been proposed. In [12], IBC-based cryptography system is used for communications in smart grid networks, where machine identification number of a device is used to generate unique keys. Not only this scheme is computationally expensive but also it requires a modification inside each measurement device (i.e., the memory of pole-top Measurement devices needs to be reconfigured) when a new data collector device is added. This approach is not feasible in the scenario that we are considering because of the limited change management capabilities [2] of the Measurement devices.

The authors in [18] propose to have a trust anchor for performing mutual authentication and key establishment between a device and a collector. This approach may not be scalable when there are many devices. In [26] authors study which unicast and multicast sessions need to be secured in a wide-area measurement system. The authors suggest keys to be established by direct connection between the two entities that need shared keys. Thus, the scheme is not suitable for secure data collection via a data collector. The work in [27] describes a key management scheme for unicast, multicast, and broadcast messages in AMI. The keys form a graph so that keys are easily stored and derived. Session keys are generated based on previously read data. Nevertheless, this scheme cannot be applied in hierarchical data collection architecture.

Another line of work deals with data collection between multiple entities in smart grid. In [20] authors describe a lightweight and scalable transport protocol for establishing multiple sessions among Measurement devices to the control center. The study focuses on how to reduce the storage needed in maintaining the state information of the massive amount of sessions. The session keys used can be derived so that the control center does not have to remember a lot of keys. Nevertheless, the number of sessions maintained is still proportional to the number of Measurement devices. The protocol is also not suitable for the hierarchical data collection architecture in which a data collector is responsible for collecting and/or processing data from multiple Measurement devices before sending the data to the control center. The authors in [19] develop a transport protocol for reporting data through a data collector. Two separate TCP connections are maintained: one between the control center and the data collector, and another between the measurement device and the data collector. Each connection can be protected independently. This approach does not consider the limited change management capabilities of Measurement devices. In addition this approach assumes that the data collector is trustworthy, which may not be always the case when the data collector is outsourced or compromised.

To the best of our knowledge, there is no existing work that allows generation of short-lived device password and keys for data collection in hierarchical data collection architecture under the stated constraints. In this thesis, we present a secure a scalable password-changing protocol for smart grid device authentication. We address the limited change management capability problem as well as the memory and computational constraint problem of Measurement devices. To the best of our knowledge, this is the very first attempt to address our goals.

There is also no such work that allows an utility/control center to generate different shared keys with different Measurement devices in a scalable manner. Existing standard protocols such as DNP3 [28] and TLS [29] are not suitable for the scenario when the data collectors are untrusted and potentially mobile with intermittent connectivity. DNP3 [28] is a standard communication protocol used for telecontrol and telemetry in SCADA systems. Its security model is not designed to provide data integrity and confidentiality against compromised relay nodes, as it assumes that all components are within the security perimeter of the operator. TLS [29], on the other hand, involves multiple phases of handshakes and is therefore not suitable if the data collector is off-line when communicating the measurement devices. In this thesis, we also present the secure and lightweight protocol that allows a power operator to collect data from measurement devices using potentially multiple mobile, non-trustworthy data collectors and analyze the lesson learned from the experiments.

CHAPTER 4

SYSTEM MODEL AND ARCHITECTURE

In this chapter we present the system model and assumptions for the framework. We present the hierarchical data collection architecture in Section 4.1. This is followed by the description of the network model that we are considering in sections 4.2. In Section 4.3 we present the data model. The user security model and possible attack scenarios are presented in Sections 4.4 and 4.5 respectively.

4.1 Architecture

Smart grid system uses a vast number of telemetric measurement devices, which are sensors with capacitor banks. These devices are placed on top of electric poles. These telemetric measurement devices usually measure frequency, voltage and current readings from power lines and store them locally. The measured telemetric data has to be delivered to a control center (i.e., utility company) securely. It is not possible to use power line communication (PLC) to deliver the measured data, since the amount of data is large. Besides, it is very expensive for each device to maintain a secure wireless connection with the control center to report data periodically. Therefore, the maintenance personnel (operators) from utility companies collect data readings from these measurement devices to their data collector devices on a regular basis to ensure that the health of power line is sound and stable.

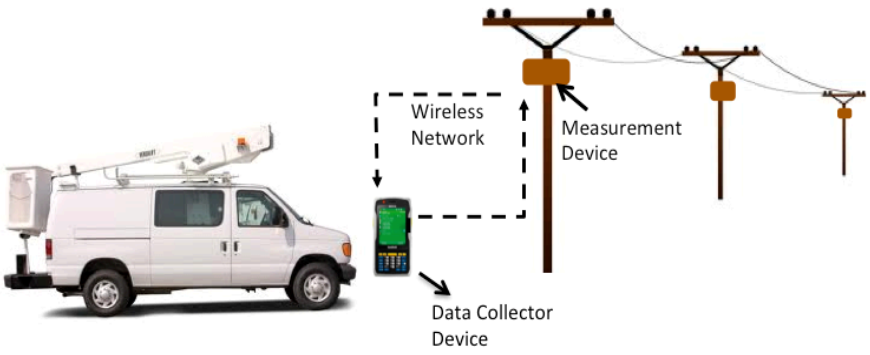

Figure 4.1: In-Field Scenario

As shown in Figure 4.1, to collect data, usually maintenance personnel carrying a data collector device drive their truck from pole to pole. After coming in the range of radio network under a pole, the operator and the data collector device need to authenticate to the measurement device. In this way the operator collect data using a data collector device from multiple measurement devices. The operator delivers the measurements to the control center when data collection from multiple measurement devices is complete.

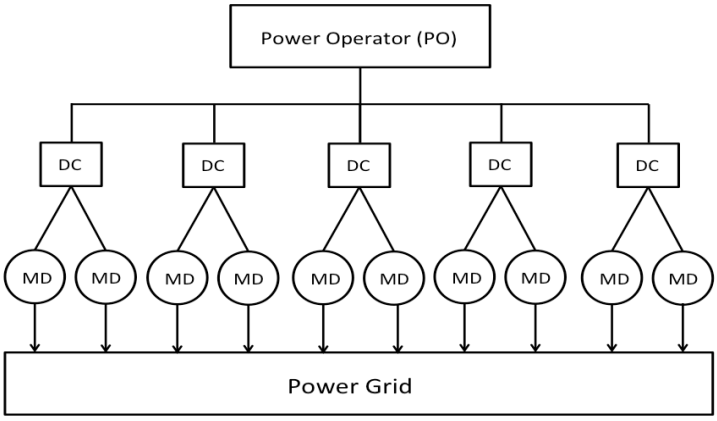

Figure 4.2: A Three Level Hierarchical Data Collection Model

Efficiency and scalability of the data collection process arguably require a hierarchical data collection framework to be adopted. Figure 4.2 shows a simplified hierarchical data collection model. Data collectors (DCs) collect data from measurement devices (MDs), and send the data to a control center, typically owned by the power operator (PO).

Table 4.1 Notations of System Principals

Symbol	Definition
OP	Operator/Maintenance Person
MD	Measurement Device
DC	Data Collector
PO	Power Operator/Control Center/Utility

Each Data collector (DC) is responsible for collecting data from multiple measurement devices (MDs), and therefore the power operator (PO) only needs to communicate with a few Data collectors (DCs) directly, keeping the number of connections small to maintain manageability. Each measurement device (MD) takes measurements from the physical infrastructural system and forwards the data to an assigned Data collector (DC). Power operator (PO) collects data from multiple Data collectors (DCs). Table 1 summarizes the notation of system principals.

4.2 Network Model

In our smart-grid setup, sensor and capacitor banks, placed on electric poles, measure telemetric measurements from power line and store it in a local memory as mentioned in the previous section (Figure 4.1). A radio attached underneath the capacitor banks is used to transfer the stored data readings. We consider two devices throughout this thesis - a measurement device (MD) that produces data measurements from the capacitor banks and a data collector device (DC) that collects these data readings. A point-to-point radio (wireless) network is established between DC and MD as communication channel. The standard we use in our validation is IEEE 802.11n. However, other wireless standards such as IEEE 802.15.4 (Zigbee) can also be used.

A secure communication channel (wired or wireless) is established within the control center's perimeter, and is used to transfer data from DCs to PO. However, extending the security perimeter to a vast number of DCs is expensive and can even be infeasible when the data collection architecture is changeable or DCs are mobile. In this thesis we consider both cases, i.e., a security perimeter is present to transfer the data readings from DCs to PO and no security perimeter is present to transfer the data readings.

4.3 Data Model

Usually, measurement devices (MDs) collect telemetric measurements of frequency, current, voltage readings of power lines and store these measurements locally. According to the utility companies [10], these telemetric measurements are not sent over PLC (Power Line Communication) because the amount of data is large. The measurements are sent from the measurement devices to the data collector devices

over the wireless network in small packets. Intruders may get unauthorized access and change the telemetric measurements maliciously at the measurement device, which may lead to wrong decision-making at the utility. Therefore, securing the access of devices (both measurement device and data collector device) and the communication channel between them is important for the utility companies.

4.4 User Security Model

We assume that the operator can only access a data collector device if s/he has a unique identification number, OP_{id} and a user password that is shared among operators. There is a trusted setup phase at the utility site prior to any communication, when OP_{id}–*password* database is stored on data collector devices. In addition, key based hash functions (e.g., SHA-2), pseudorandom generator function, necessary crypto algorithms such as symmetric key algorithm (e.g., AES) and public-key encryption algorithm (e.g., RSA) are agreed upon and installed on utility, data collector and measurement device. The installation and update configuration of functions/keys on measurement device are critical and out of the current scope of the thesis. Note that we do not include integrity check of data readings by data collector and/or utility in the first part of the thesis (Chapter 5). Also, we do not include human operator authentication in the second part of our thesis (Chapter 6). However, if the utility requires, human operator authentication step can be included before any communication between data collector and measurement device.

For the SCAPACH protocol, we use both symmetric and public-key encryption algorithms for protocol message communication over the wireless network, and symmetric-key encryption algorithm for telemetric data readings. For key establishment in our SELINDA framework, the Diffie-Hellman (DH) mechanism [32] is adopted, which is discussed in Section 2.2. In addition, public-key encryption algorithms are used for communication over the wireless network.

4.5 Attack Model

Since the whole communication system exists in an open environment, security barriers to prevent unauthorized access by potential eavesdropper or active attacker are extremely necessary. In this thesis, we only consider cyber-security attacks. Physical attacks and security protections against them

are out of scope of this thesis. An attacker may try to get access of the devices by faking identities if the attacker gets the shared user password. Besides, since the network is wireless, attacker may eavesdrop on the communications and place man-in-the-middle attack on-site or a replay attack at later time. Even worse attackers may get access to measurement device, break cryptographic keys information and falsify/tamper with the telemetric data. Also, the confidentiality of the data sent in earlier sessions may be lost if the long-term secrets are obtained by the attacker at a later time.

CHAPTER 5

SCAPACH - Scalable Password-Changing Protocol for Smart Grid Device Authentication

In this section we present a SCalable and Automated PAssword-CHanging protocol, SCAPACH [37], for unique authentication of human personnel (operator) and secure collection of telemetric data readings from a large number of measurement devices. The problem description is stated in Section 5.1. This is followed by a description of the assumptions in Section 5.2 and detailed protocol in Section 5.3. In Section 5.4 we analyze the security of the SCAPACH protocol. The implementation details are presented in Section 5.5 followed by the evaluation in Section 5.6.

5.1 Problem Description

The overall goal is to ensure unique and secure authentication of human personnel (operator) and secure collection of telemetric data from a large number of measurement devices in real-time under varying maintenance scenarios. In particular our focus is on delivering the telemetric measurement in a secure and fast manner under the **resource constraints** (in terms of both memory and computation), **lengthy deployment** and **change management capacity** of measurement devices. We also need to ensure that our protocol performs efficiently with the large **scale** of measurement devices and data collector devices. Moreover, our focus is to keep the protocol **computationally lightweight** for the measurement devices so that they require to perform very few expensive cryptographic operations and to exchange few messages.

The problem is to generate unique passwords and symmetric keys for authentication and encryption respectively, when operators $(OP_1, OP_2, \ldots, OP_i)$ use data collector devices (DC_1, DC_2,…, DC_i) to collect data from measurement devices (MD_1, MD_2, …, MD_j) at different locations (L_1, L_2, …, L_j) at different times (TS_1, TS_2, …, TS_j).

5.2 Assumptions

Utilities deal with a large number of MDs. Operators collect the measurements from MD using DCs. Multiple operators may use the same DC at different days to collect telemetric data. On a particular day, operators $(OP_1, OP_2, \ldots, OP_i)$ use data collector devices $(DC_1, DC_2, \ldots, DC_i)$ to collect data from measurement devices $(MD_1, MD_2, \ldots, MD_j)$ at different locations $(L_1, L_2, \ldots, L_j)$ at different times $(TS_1, TS_2, \ldots, TS_j)$.

For maintaining confidentiality and authenticity of initial setup messages, public-private key pairs (PU_j-PR_j) are defined for each MD, and stored inside the DCs. However, since MDs are memory-constrained devices, instead of storing public keys of all DCs, we generate on-the-fly symmetric keys using introduce *Physical Unclonable Functions (PUFs)* [3] attached with MDs to ensure a key agreement between DC and MD. This symmetric key is only used for securing initial protocol messages between them. Volatile cryptographic key generation using PUF is discussed in Section 2.4.

We assume that the PUF system-on-chip (SoC) is integrated with each MD. During the trusted setup phase at the utility site, the utility constructs CRPs for the PUFs inside each MD, which is also stored into DCs' databases.

We assume that MD_j only stores its own private key (PR_j) and a shared secret with DCs in form of salt ($S_{cur,j} < 1$) in its firmware. On the other hand, DC_i has a list of public keys (PU) of all MDs in its memory in addition to all CRPs associated for all PUFs (in MDs). DC also stores a list of shared secrets, i.e., salts ($S_{cur,1}, S_{cur,2}, \ldots, S_{cur,j}$) in its firmware. In addition, a database of OP_{id}-*password* of all operators is stored in DC for human (operator) authentication. Both devices have the capability to execute AES, RSA, SHA-2 cryptographic algorithms and functions (defined in the following sections) to generate device passwords and one-time shared keys (P).

5.3 Protocol

In this section, we present our password-changing protocol, SCAPACH that provides robust authentication and secure communication. We divide our approach into three phases:

- Phase 1: Authentication of an operator (OP_k) to the data collector device (DC_i)

- Phase 2: Authentication between the data collector device (DC_i) and the measurement device (MD_j)
- Phase 3: Secure communication between the data collector device (DC_i) and the measurement device (MD_j).

The functionalities of the data collector device are built into the utility car. So, the set of activities on phase 1, i.e., authentication of the operator to the data collector device, needs to be once when s/he starts driving for collecting data, and not at each pole. Moreover, operator authenticates his/her data collector device to each measurement device with a unique device password at each pole location to collect data readings (phase 2 and 3). Detailed steps of these phases are described as follows. Mathematical notations of the symbols used in this section are given in Table 5.1.

Table 5.1 Mathematical Notations

Symbol	Definition
$E_{PUj}()$	Encrypt operation with Public Key of j^{th} MD
$D_{PRj}()$	Decrypt operation with Private Key of j^{th} MD
$[M]_{PR}$	Sign a message *M* with own private key
$E_P()/D_p()$	Encrypt/Decrypt with symmetric key, *P*
$E_{KC}()/D_{KC}()$	Encrypt/Decrypt with symmetric key *KC*
P	Session shared key of 256 bit
KC	Symmetric key generated by PUF
$p^{/}$	*k* bits of *P* starting from index *n*
k	Number of bits of *P* to verify
Ch_k^j	Challenge for PUF associated with j^{th} MD chosen from a set of *k* challenges
$S_{cur,j}$, $S_{prev,j}$	Salt (current and previous) at j^{th} MD
L	Location
TS	Time variant nonce
nonce	Random number
DC_{id}	Data collector Device id - 48bit MAC address
OP_{id}	Operator Identification number
ACK	Acknowledgement
ERR	Error message
TER	Terminate message
f()	Pseudorandom generator function
Q()	256-bit cryptographic hash function
||	Append Operation

A. ***Phase 1.*** All methods of human authentication fall into three broad categories [6]:

- The knowledge factors: Something the user knows (e.g., password, pass phrase, PIN, response to a challenge)
- The ownership factors: Something the user has (e.g., ID card, security token, software token, cell phone)
- The inherence factors: Something the user is or does (e.g., fingerprint, retinal pattern, DNA sequence, signature, face, voice)

In our approach, we consider the first category since it is easier to use, convenient and less expensive to deploy than token-based or biometric methods. This phase deals with the authentication of operator to data collector device, yet maintaining the integrity of operator. To authenticate, OP_k provides a valid unique user identification number (OP_{id}) and shared user password to DC_i. The step ensures the identification and authentication of an operator. However, remote software robots may try to get access of MD by breaking into DC. To protect that, a CAPTCHA test [7] is introduced. DC generates a CAPTCHA using cyber-physical information (i.e., GPS location, temperature, and data collector device id DC_{id}), which is collected at the beginning of phase 1 using respective sensors.

OP_k	DC_i
	1.1 Generate a CAPTCHA
1.2. Enter Answer of CAPTCHA	
	1.3 Verify CAPTCHA Answer
2.1. Enter OP_{id} and Password	
	2.2 Verify OP_{id}, Password

Figure 5.1: Phase 1 - Authentication of Operator

Figure 5.1 formalizes the procedure of a robust authentication of OP in phase 1. Authentication of OP is important so that responsible OP can be identified in case of an insider attack. After OP authenticates, DC sends the login request message to MD (in the next phase). Phase 1 is associated with a timer or counter. When the counter expires (e.g., after few hours or visiting few different locations), the OP needs to perform re-authentication.

B. *Phase 2.* In this phase, DC authenticates itself to the MD and calculates the session-shared keys (to be used for transmitting telemetric data). As soon as the OP comes within the range of MD's wireless network, DC combines OP_{id}, DC_{id} (collected in phase 1) and time variant nonce *TS* in the form of a **login request message** m_1. DC then chooses a challenge-response pair, Ch_k^j-Rs_k^j from a set of *k* CRPs stored for j^{th} MD, and generates a key *KC* by hashing Rs_k^j (using hash function H_1:{0,1} ^{t} → {0,1} ^{m}). DC encrypts message m_1 with *KC* and appends Ch_k^j so that MD can regenerate key *KC* from Ch_k^j using the PUF SoC (section IIE). Thus a volatile key, *KC* is agreed between MD and DC without storing additional keys in MD.

The DC then encrypts again with the public key of MD and transmits the encrypted message (c_1 in Figure 5.2) to MD over the wireless network to initiate a conversation with the MD. Note that we do not require any clock synchronization between MD and DC.

When MD receives m_1, it extracts challenge Ch_k^j by decrypting c_1 using its own private key PR_j, generates key *KC* from PUF (using Ch_k^j), decrypts the rest of the message with *KC* and identifies OP_{id}, *TS* and DC_{id}. MD generates a random *nonce, k* and *n,* where both *k* and *n* are chosen from a range of numbers. A message m_2 is constructed by appending *nonce, k, n,* and extracted *TS* (from m_1) together. Then m_2 is transmitted to DC after encrypting with *KC* and signing with MD's private key to ensure the confidentiality and authenticity of the message (c_2 in Figure 5.2).

Next, both devices (MD and DC) start calculating the same *P* using the equation: $P = Q\ (OP_{id}, S_{cur,j}, nonce)$. Here *P* is a symmetric key used in the final phase for en/decrypting telemetric data and *Q()* is a 256-bit cryptographic hash function, e.g., SHA-2.

In function *Q()*, we use a salt value, $S_{cur,j}$ (<1) as an input, which is calculated using a pseudorandom generator function *f()* with seed [$S_{prev,j}$ || DC_{id} || *TS*]. At every session, a new $S_{cur,j}$ is calculated, which becomes $S_{prev,j}$ at the end of the session to be used for the next session. This way, value of *salt* changes for every session based on a secret value ($S_{prev,j}$) stored in the firmware of both MD and DC (installed beforehand). Hence, it is hard for the attacker to guess the value of $S_{cur,j}$.

Note that $S_{cur,j}$ values vary across MDs. Once $S_{cur,j}$ is assigned as $S_{prev,j}$ for future computations, the updated $S_{prev,j}$ needs to be disseminated to other DCs before they access the same MD. The synchronization of $S_{prev,j}$ across the DCs is done at the end-of the day at the utilities. Since one

MD is accessed maximum once a day, synchronization of $S_{prev,j}$ at the utilities does not require any behavioral changes in the measurement.

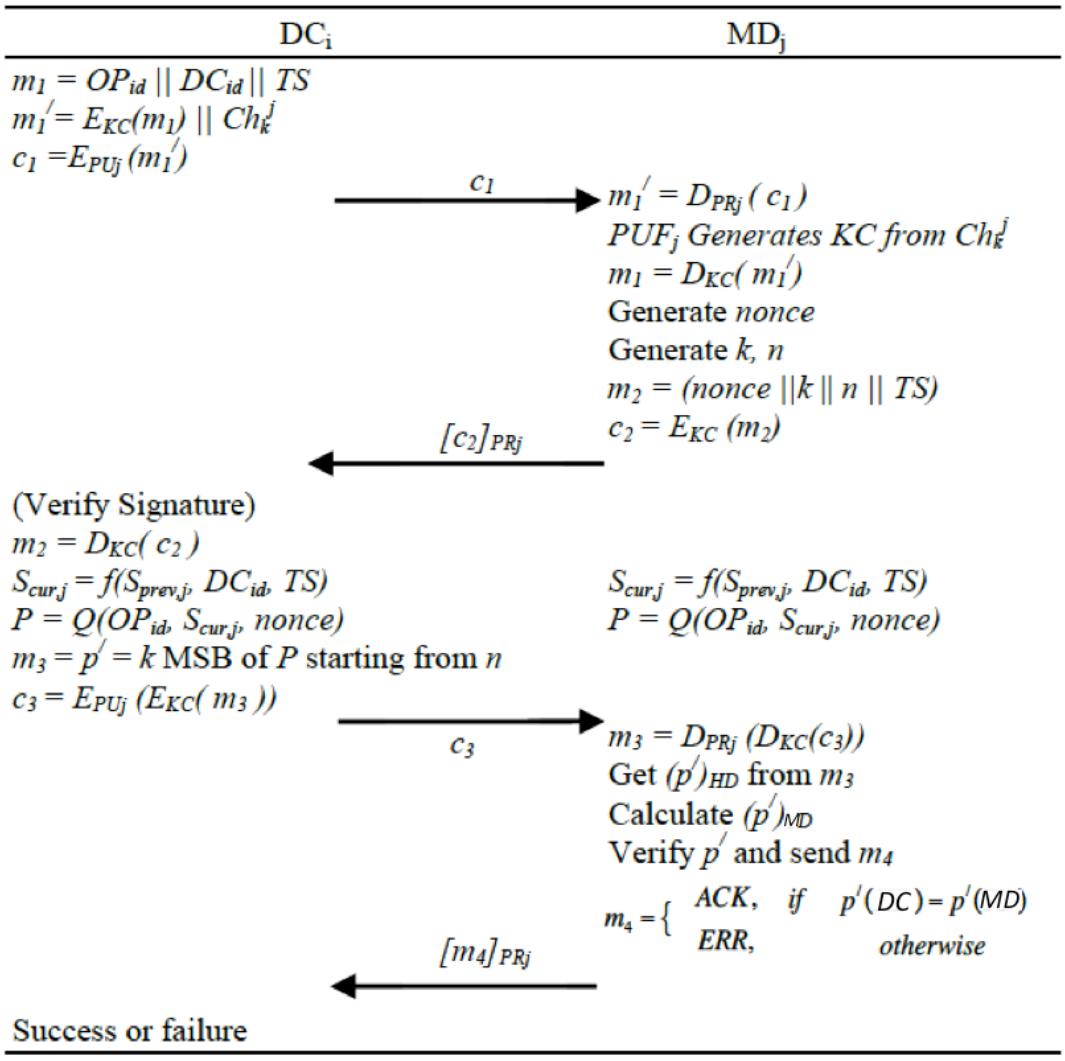

Figure 5.2: Phase 2 - Authentication of DC to MD and Shared Key Generation

In our password changing protocol, only *k* bits from index *n* of shared-symmetric key *P* are used as device password (p') for the authentication of DC to MD (different from the OP's shared password entered in phase 1). DC picks p' (message m_3 in Figure 5.2), encrypts it with *KC* and public key of MD and transmits the encrypted message c_3 to the MD as the computed response. Upon receiving c_3, MD decrypts it and extracts p' calculated by DC. The MD then validates

received p' with the self-computed p'. If the received and local p' values do not match with each other, the authentication is failed and an error message, *ERR* is sent. Otherwise, an acknowledgement, *ACK* is sent to the DC.

Message m_4 is also sent after signing it with the private key, so that the DC knows that this message is coming from a legitimate MD. $S_{prev,j}$ is updated only when an authentication is successful. The authenticity and confidentiality of the messages is maintained by using both *KC* and the private key (PR_i) of MD. Note that *KC* is not used as the symmetric key for telemetric data encryption in phase 3, since *KC* repeats for the same input Ch_i, which invalidates the notion of one-time password and key generation. Therefore, one-time key *P* is generated in this phase.

C. ***Phase 3.*** In this phase, secure delivery of the telemetric data is ensured. Both devices use the *256* bits *P* derived in phase 2 as the symmetric key. The MD_j reads the telemetric measurements from memory, encrypts the data with *P,* signs and sends it to DC_i over the wireless network. Upon receiving the data DC_i stores them in secure database. Finally, they conclude when MD_j sends a signed termination message to ensure that the session is terminated. Figure 5.3 shows the details.

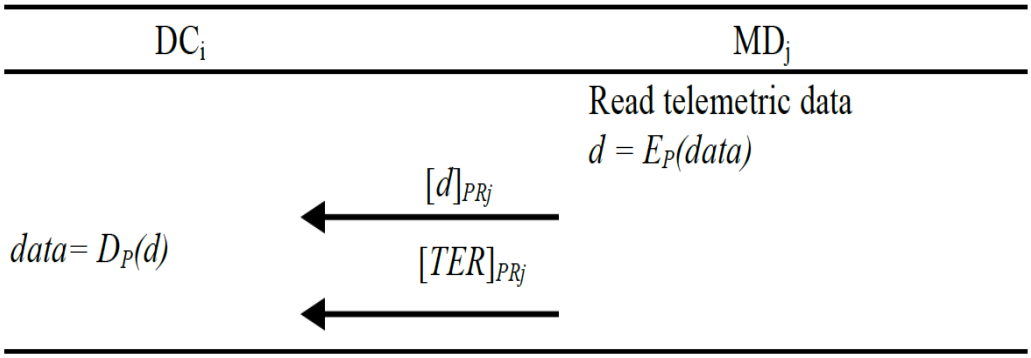

Figure 5.3: Phase 3 - Communication Between Two Devices

5.4 Security Analysis

In this section, we analyze the security of SCAPACH against various cyber-attacks that are considered important to address in literature [9][13].

- ***Replay Attack:***

  In SCAPACH, the device password keeps changing across each data collection session. Therefore, even if an attacker eavesdrops the flying message (c_3), she cannot use it for future sessions to authenticate. Moreover, to protect against replay of c_1, an alternative of our protocol can be formulated. While constructing $m_1^{/}$ (as shown in Figure 5.2), DC can encrypt m_1 separately with secret $S_{cur,j}$ and *KC*, and send both of them in $m_1^{/}$. Since $S_{cur,j}$ is different for each session, MD can check whether c_1 is intended for current session or not by verifying both m_1. Thus our protocol thwarts replay attack. This alternative also thwarts **Denial of Service (DoS)** attacks. However, this alternative introduces another level of decryption and hence, there is a tradeoff between the computational cost and security against the DoS attack.

- ***Perfect Forward Secrecy:***

  The forward secrecy property ensures that the conversation an adversary recorded remains secret if one of the private keys is compromised in the future. In our protocol, even if an intruder gets access to private key of MD, she cannot derive the messages exchanged. It is because, m_1, m_2, m_3 are encrypted using *KC*, which changes depending on input challenges. Therefore, even if the private key is compromised, attacker cannot derive *P* and hence SCAPACH maintains perfect forward secrecy.

- ***MITM Attack:***

  A man-in-the-middle (MITM) attack requires an attacker to fool both sides of a legitimate conversation [5]. This is not possible in our protocol since a key agreement needs to be established between DC and MD (more discussed below).

- ***Masquerade of Measurement Device:***

  All messages sent from DC are encrypted with the public key of MD. To protect the masquerade of MD, DC sends time variant nonce (*TS*) that MD needs to send back in m_2. DC makes sure that it is

talking to a legitimate MD by verifying the received *TS* in c_2. Because, attacker does not have the private key of MD and so they cannot decrypt and reveal correct *TS* from c_1. Also, MD signs the messages with its private key PR_j, which also ensures DC that it is communicating with a legitimate MD.

- **Masquerade of Data collector Device:**

  An intruder can generate a garbled c_1 and send it to MD to pretend like DC. The MD extracts *Ch* from received message and the associated PUF generates the key *KC* (using garbled Ch), which both parties need to use for further communication. However, according to the property of PUF, an attacker can never produce correct *Rs* from a given *Ch* [14]. PUFs can only be broken by numerical modeling attacks if the attacker knows a set of CRPs of a PUF [15]. However, no CRP is revealed during any communication in our protocol. So, the attacker can never derive correct *KC* and hence cannot decrypt c_2. Moreover, MD sends the nonce that is supposed to be used as an input to calculate *P* only for that session. Therefore, the intruder can never compute a valid *P* and hence cannot pretend to be an authorized DC.

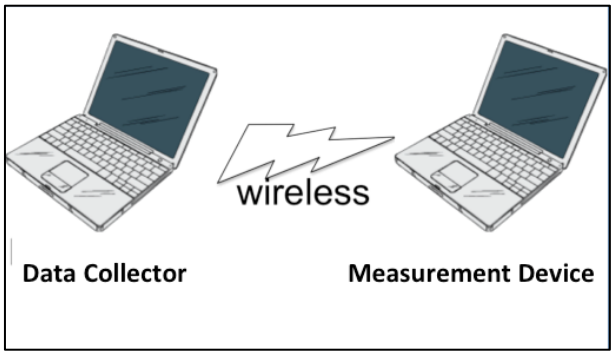

Figure 5.4: Experimental Setup to Measure the Performance of SCAPCH Protocol

5.5 Implementation

To validate the SCAPACH protocol, we use two laptops as DC and MD with Intel Core 2 Duo 2.26GHz processor and 2GB read-only memory. The prototype is implemented in Java so that it can be easily ported into mobile phone-like devices. The communication between laptops uses wifi 802.11n wireless

network. RSA is used as public key encryption. Besides, AES is used as symmetric key algorithm to encrypt telemetric readings, since it is faster for larger size of data. Figure 5.4 shows the experimental setup.

5.6 Evaluation

To compute the performance of SCAPACH, we measure its total execution time, which is 730ms on average. Figure 5.5 shows the CDF of execution times of SCAPACH in three phases over 100 executions. The authentication process of operator in Phase 1 is done locally at DC. The operator-side delay in this authentication process is considered negligible. Therefore, the execution time in Phase 1 is very small (average 26.7ms).

Phase 2 takes the highest time due to the repeated communication between MD and DC. However, this execution time is less than 600ms in most of the cases (80% cases in Fig. 5.5). The execution time in Phase 3 is about 150ms on average.

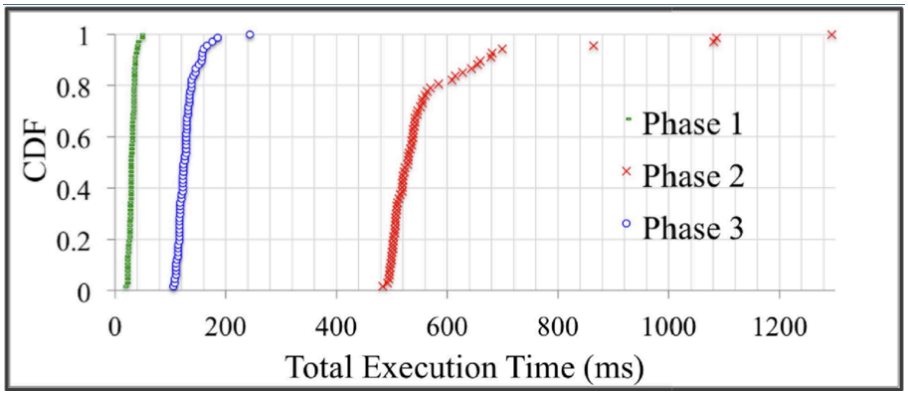

Figure 5.5: Execution Time of SCAPATH Over 100 Executions

Note that we do not consider the computational time of *PUF* here since it is a separate SoC and can generate keys in parallel with MD. The network communication delay is about 15-20ms. We also measure the execution time of different processes (e.g., encryption, computation of *P*, etc.) inside each phase. We find that the RSA encryption-decryption time is the main contributor to the execution time in

phase 2 and phase 3. We run our experiment at different time of the day to get a more accurate result, since the network delay varies at different time of the day. Our implementation results indicate that SCAPACH works efficiently.

CHAPTER 6

SELINDA- Secure, Scalable and Lightweight Data Collection Protocol for Smart Grids

In this section we introduce a Secure, Scalable and Lightweight Data Collection Protocol for Smart Grids that allows utility company to collect telemetric data readings from a large number of measurement devices via mobile and untrusted data collector. The problem description and the description of the assumptions are stated in Section 6.1 and Section 6.2 respectively. The detailed protocol is discussed in Section 6.3. In Section 6.4 we describe the protocol complexity and security of the SELINDA protocol. The implementation details are presented in Section 6.5 followed by the evaluation in Section 6.6.

6.1 Problem Description

Security in the smart grid is a challenge as an increasing number of sensors and measurement devices are connected to the power grid. General-purpose security protocols are not suitable for providing data security to devices with limited memory, computational power and network connectivity. In this work, the goal is to develop a secure and lightweight scalable security protocol. The protocol will allow a PO to establish shared keys with multiple MDs via an untrusted DC. The DC behaves like a relay for data communications although it is not continuously connected to the PO. Besides, the DC has no access to the keys established between the PO and the MDs. Therefore, the DC can potentially be mobile and untrusted, which makes the scheme essential for ensuring the security of community aided data collection in the smart grid.

SELINDA [1] protocol has four key features that distinguish it from existing protocols. First, the protocol is computationally lightweight for the MDs since it requires the MDs to perform very few expensive cryptographic operations and to send few messages. Thus the protocol supports resource constrained MDs with limited memory and a slow CPU. Second, the protocol allows to trade off computation for memory requirements in the MDs. There is one long-term secret per entity, its private key. Thus, the PO needs to maintain the public key of all MDs in the system. It also maintains state information for the

current session of data collection. The MD only needs to remember its own private key and the public key of the PO. They can recalculate or store the session keys, depending on their computational and memory constraints. Third, the protocol provides flexible association between DCs and MDs. As an MD does not need to know the public key of any DC, the PO can assign different DCs to collect data from the same MD at different times. This feature is particularly important in scenarios where the DCs are mobile or the infrastructure is evolving so that different mappings or assignments between DCs and MDs can be used at different times after the deployment of the MDs. Finally, the protocol protects the collected data from a compromised DC. Thus, an attacker cannot access the collected data even if it is in control of a DC. This allows the PO to outsource the data collection procedure without sacrificing security.

6.2 Assumptions

We assume that the PO, the DCs, and the MDs are initially configured with the following set of parameters:

Long-term keys: The PO, every DC, and every MD have their public and private key pairs; the PO knows the public keys of all MDs and DCs, while the MDs know the public key of the PO. MD does not store the public key of DC so that PO can flexibly assign different DCs to collect data at different time.

Table 6.1 Notations

Symbol	Definition
K{M}	Encrypt message *M* using shared key *K*
[M]A	Sign message *M* using the private key of entity *A*
{M}A	Encrypt message *M* using the public key of entity *A*
G(p)	Multiplicative group over prime *p*
$G_q(p)$	$G_q(p)$ Subgroup of *G(p)* of order *q*
g	Generator of subgroup $G_{q0}(p)$ for a large prime q_0

Diffie-Hellman (DH) parameters: The PO and the MDs agree on parameters of the prime order digital signature algorithm subgroup $G_{q0}(p)$ with generator g of group $G(p)$. The length of p is the same order of magnitude as that of a public key, but q_0 is substantially smaller.

Cryptographic functions: The PO and the MDs have a common set of cryptographic schemes for encryption (e.g., AES) and for hash computation (e.g., SHA-256). We use a keyed hash for data authentication. In the rest of this chapter, we call the key for encrypting data the encryption key, and the key for providing integrity the integrity key.

In total, every MD has to store one private key, one public key, and the parameters of the group $G_{q0}(p)$, i.e., g, q_0 and p.

6.3 Protocol

Following the hierarchical data collection model, we describe the SELINDA protocol based on the two sessions that it consists of: the PO-DC session and the DC-MD session. Figure 6.1 illustrates the complete data collection process. Table 6.1 defines the notations used in the figure.

A. PO-DC Session

The purpose of the PO-DC session is to provide to DC the information needed to be able to collect the data: the list of MDs that the DC has to collect data from, the public keys of the MDs and the DH half keys. The DH half keys will be used later for establishing the encryption/integrity keys between the PO and the MD, and between the DC and the MD. The session is composed of the following messages between the PO and the DC.

(1) PO to DC: $[\{g^a \bmod p \,||\, T1\}_{DC}]_{PO}$

$T1$ is the current timestamp of PO. DC checks whether $T1$ is close to its own time. To detect replay attacks, DC should keep the previous $T1$ received. The current $T1$ is accepted only when it is later than the previous one.

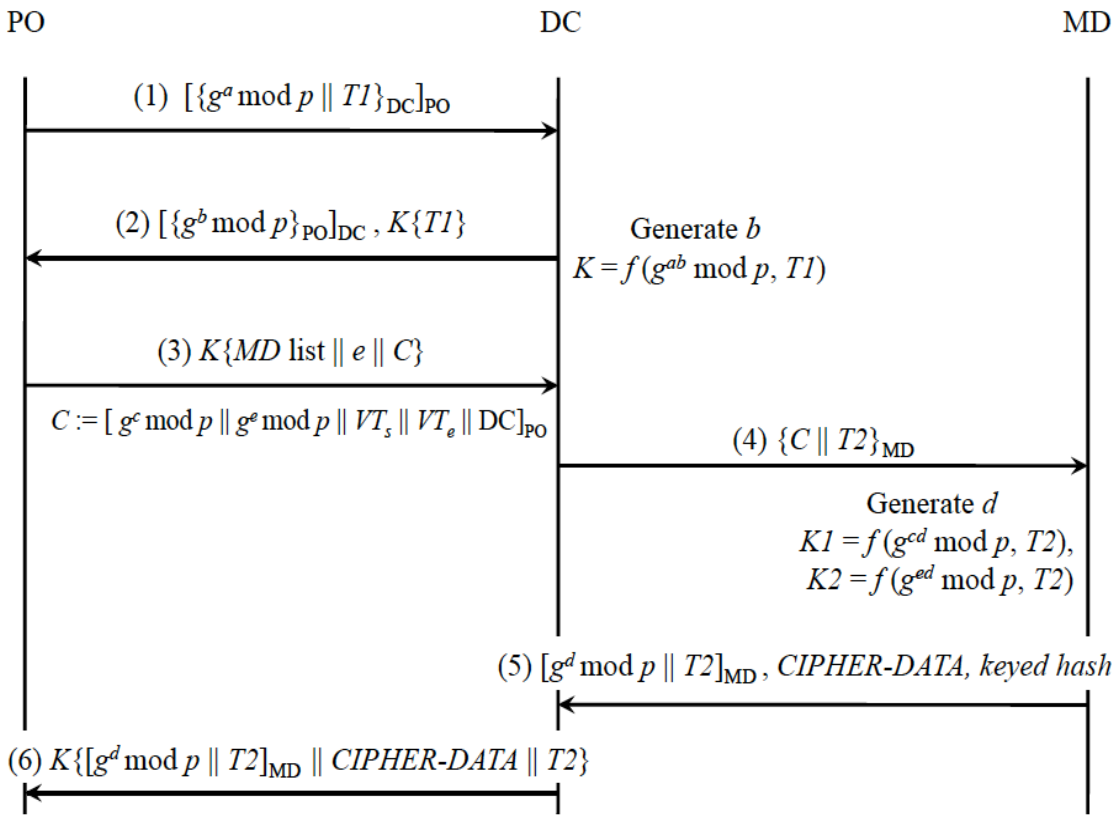

Figure 6.1: The SELINDA Protocol

(2) DC to PO: [{ g^b *mod p* }$_{PO}$]$_{DC}$, *K*{*T1*}

If *T1* is valid, DC generates its DH half key g^b *mod p* and computes the shared key *K*. *K* depends on g^{ab} *mod p*, *T1*, and the identity of DC. Recall that both the DC and the PO know the function to generate *K*, thus the PO can generate *K* after receiving g^b *mod p*. DC sends *K*{*T1*} to authenticate itself to the PO, which the PO does by verifying *T1* carried in *K*{*T1*}. If the authentication is successful, the PO can send to the DC the list of MDs, together with the public key information of the MDs.

(3) PO to DC: *K*{ MD list || *e* || *C* }, where *C* = [g^c *mod p* || g^e *mod p* || VT_s || VT_e || *DC*]$_{PO}$

C is called "token" from PO to DC for data collection.

The token, and the DH half keys g^c *mod p* and g^e *mod p* inside, will be valid during the time period [VT_s,VT_e], and VT_s must be later than *T1*. g^c *mod p* is used for establishing shared keys

between PO and MDs. $g^e \bmod p$, on the other hand, is for DC to establish a session key with the MDs. Note that e is contained in the message so that DC can retrieve it using K. It is important to note the fact that $g^e \bmod p$ is provided by the PO assures that a compromised DC cannot perform a small subgroup attack on the MDs' DH keys.

To be able to decrypt the data collected by the DCs, the PO has to remember one *K* per DC, but DH secret *c* can be the same for all DCs. Therefore, the memory requirement at the PO is very low. To further reduce the state information kept at the PO, it is possible to close the session after the DC has received the token, and resume the session when the DC is about to report data. Whether or not this is done does not affect the rest of the protocol.

B. DC-MD Session

The DC-MD session happens when the DC is able to communicate to one of the MDs, and its purpose is the actual data collection. It contains two messages between the DC and every MD, and one message from the DC to the PO.

(4) DC to MD: $\{C, T2\}_{MD}$ where $C = [g^c \bmod p \,||\, g^e \bmod p \,||\, VT_s \,||\, VT_e \,||\, DC]_{PO}$

T2 is the current timestamp of the DC, and *C* is the token received in Step (3). When the MD receives the message, it verifies it by checking whether *T2* is close to its current time and whether *T2* falls in the range $[VT_s, VT_e]$. *T2* should be also later than the previous timestamp used for the same purpose. The MD then generates its reply.

(5) MD to DC: $[\{g^d \bmod p\}_{PO} || T2]_{MD}$, *CIPHER-DATA*, *keyed hash*

Let DATA be the data measured by the MD in plaintext, i.e., the data to be collected. As mentioned earlier, we want to encrypt *DATA* in a way that only the PO can read it, but at the same time, we want to allow DC to check the integrity of DATA, so it can detect if an attacker between MD and DC tampers with the data. We achieve this by establishing one session key between the PO and the MD, and one session key between the DC and the MD. Both keys are established through DH, and to reduce complexity, the MD uses the same DH public key ($g^d \bmod p$) for both keys.

Given *d* (MD's DH private key), the MD can derive the session key *K1* shared with the PO based on $g^{cd} \bmod p$ and based on *T2*. By using *T2* to establish *K1*, even if the DH half keys are reused to save computation, *K1* will be different in every DC-MD session. The MD can then use a standard mechanism to develop the encryption key and the integrity key from *K1*, e.g., using a hash function to hash *K1* with other information as done in IPSec. Similarly, the MD obtains *K2* based on $g^{ed} \bmod p$ and *T2*.

The MD then uses the keys derived from *K1* to encrypt and to authenticate DATA towards the PO; the result is denoted by CIPHER-DATA. Observe that CIPHER-DATA is piggybacked in the same message as the DH public key used to generate the session key *K1*. As we show later, this is important for security. Finally, the MD uses *K2* as a key to generate a keyed hash of CIPHER-DATA.

Upon receiving the message from the MD, the DC verifies *T2* from the signed message. DC can then compute *K2* using $g^d \bmod p$ and *T2*, and can verify the integrity of CIPHER- DATA. If the integrity check is successful, DC encrypts CIPHER-DATA and *T2* using the session key *K* it established with the PO in the PO-DC session, and sends it to the PO. The PO uses $g^d \bmod p$ and *T2* to compute *K1* and the encryption and integrity keys needed to decrypt and to validate CIPHER- DATA. The data collection process is complete once the PO receives and validates DATA.

6.4 Security and Complexity Analysis

- **Protocol Complexity:**

We study the number of expensive operations the MD has to perform every time data are collected. In practice, both public key operations and DH operations are regarded as expensive, while shared key operations and hash operations are not. Upon receiving Message 4 in Figure 6.1, MD has to perform a public key decryption to retrieve *C* and *T2*. To verify it is PO who has signed C, a signature verification operation is needed. Three DH operations are needed to generate $g^d \bmod p$, $g^{cd} \bmod p$, and $g^{ed} \bmod p$. Finally, a signature operation is needed to sign $g^d \bmod p$ and *T2* in Message 5. There are altogether three public key operations and three DH operations.

It is worth noting that the DH operations become unnecessary if we reuse the DH keys. Suppose after the first data collection, MD keeps $g^d \bmod p$, $g^{cd} \bmod p$, and $g^{ed} \bmod p$ in its memory. PO also keeps c and e. In the next collection, PO can send ["*REUSE DH* "|| VT_s || VT_e || *DC*]$_{PO}$ as a token to DC. When MD receives the token, it does not have to generate a new *d* or recompute $g^{cd} \bmod p$ and $g^{ed} \bmod p$. Note that because *T2* in the new session must be different from the last one, *K1* and *K2* will be different from the last session even though the DH keys are the same. In principle, reusing c several times could make it easier for an attacker to guess *c*. In the following, we show that for SELINDA this is not true.

- **Security Analysis:**

In SELINDA, a DC checks whether the value of received *T1* is greater than previous stored *T1*. In addition, upon receiving message 4, MD checks whether the received *T2* value is greater than previously stored *T2* and whether $VT_s < T2 < VT_e$. Thus a replay attack cannot be possible.

In the protocol, DC checks the integrity of data and it can detect if an attacker between MD and DC tampers with the data. To achieve this a shared session key is established between the DC and the MD. Besides, a *keyed-hash* of CIPHER-DATA is attached in message 5. By verifying the hash DC ensures that the data is not tampered.

To reduce the memory needed to maintain key information, the PO uses the same DH public key to develop different shared keys with different MDs. This approach may be subject to the *small subgroup attack,* i.e., an attack in which several compromised MDs try to guess the PO's DH secret key. However, it does not make the protocol subject to simultaneous small subgroup attacks because of the fact that MDs have to *piggyback* data with their public keys in Message 5. Detail security analysis is provided in [1].

SELINDA protocol also protects the confidentiality of the data sent in earlier sessions even if the attacker obtains the long-term secrets later. The Diffie-Hellman (DH) mechanism is adopted for key establishment to support this *Perfect Forward Security* property.

6.5 Implementation

To further understand the computational performance of the SELINDA protocol, we measure the time needed for DC to collect data from MD. The MD and DC are simulated using two laptops with Intel Core i5 2.4GHz processor and 4GB read-only memory. Figure 6.2 shows the experimental setup. The prototype is implemented in Java. The laptops communicate through a Wi-Fi 802.11n wireless network. We use RSA with key length 2048 bits as the public-key cryptography and AES with key length 256 bits for symmetric key cryptography. We use SHA-256 to compute hash-based message authentication (HMAC), and use Diffie- Hellman with a prime order of length 1024 bits.

Figure 6.2: Experimental Setup to Measure the Performance of SELINDA Protocol

6.6 Evaluation

To understand the computational complexity on the MDs, we have measured the time needed to decrypt Message 4 and create Message 5 (c.f., Figure 6.1). We have also measured the total time needed for the DC to collect data from a MD in the DC-MD session, which includes the time it takes to create Message 4, the round-trip network delay to send and receive Messages 4 and 5, as well as the time required to verify the integrity of CIPHER-DATA. Figure 6.3 shows the measurement results as a function of the size of DATA. Each data point is the average time of 30 different trials. We run the trials at different times of the day to get a more accurate result, since the network delay varies from time to time. For each data size, 10 trials are run at morning (medium network delay), 10 trials are run at mid-

day (peak network delay) and 10 trials are run at late night (low network delay). We show the 95 percent confidence intervals.

For MD, the difference in computational time between data sizes of 1 byte and 4096 bytes is less than 60ms, and the difference between 1K and 4K is less than 1ms. We can conclude that our mechanism scales well for typical amounts of data to be reported in the considered smart grid context. The average total times it takes for the DC to collect data of sizes 1 byte and 4K bytes are 1241ms and 1324ms, respectively.

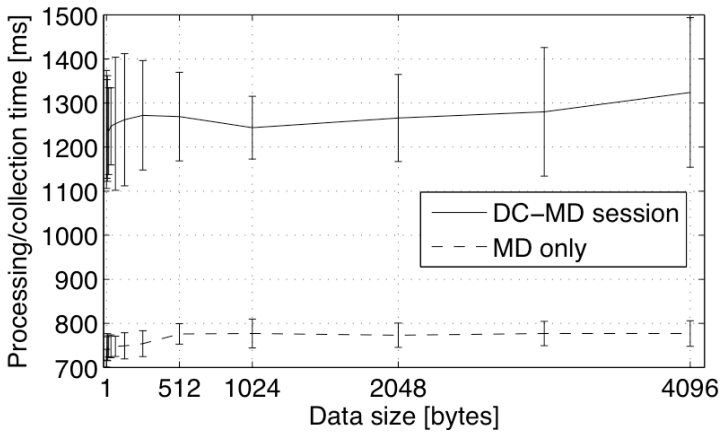

Figure 6.3: Time Performance: total time of computation in MD and total time for data collection for messages of different sizes

These results help to explore feasible mobility patterns for a mobile DC in a scenario where an automobile is used as a DC. An automobile moving at 50km/h can advance less than 20m in this amount of time. Since the communication range of 802.11 is around 250m, even a mobile DC should have enough time to complete the data collection process as long as it is not very far from the MD. We thus conclude that the proposed SELINDA protocol fulfills the goal of being scalable and efficient.

CHAPTER 7

CONCLUSION

7.1 Thesis Achievement

In this thesis, we highlight one of the realistic authentication problems in the smart grid critical infrastructure. We propose a secure authentication and data transmission protocol for collecting telemetric data from the pole devices. Our password-changing framework, SCAPACH, creates short-lived passwords and shared keys based on physical characteristics (such as per-pole device locality, data collection timestamp and per-driver identification) and changeable secrets, and ensures secure data collection considering the resource limitations of the measurement devices. The protocol is fast and secured against different security attacks in this domain.

In this thesis, we additionally discuss a secure, scalable, and lightweight protocol for smart grid data collection. This protocol allows a measurement device to report data securely to the power operator via a data collector that may not be trustworthy. It is thus suitable for data collection using mobile data collectors, and can be used for community-aided data collection. We implemented the protocol and provided measurement results that show that the protocol indeed has low computational complexity and makes mobile smart grid data collection possible.

7.2 Future Work

A lot of future work remains in this space too. First of all, we will consider how to improve the scalability and efficiency of the SCAPACH protocol. In SCAPACH protocol, we consider every measurement device holds a secret. It would be interesting to see how much the performance is influenced, in terms of security, if groups of measuring devices share a secret. In addition, different ways of forming the groups in this particular scenario can also be explored and compared to find out which type of group formation gives the best performance.

Besides, we have not considered any data aggregation by data collector device in our protocols. Data

aggregation can be included with our framework, since there are existing applications where data is aggregated before reporting the data. It would be interesting to explore if efficiency can be improved, while keeping security intact when introducing data aggregation.

Presently we run our experiments on laptop over LAN, which is a controlled environment. It would be interesting to see the performance metrics like latency, data delivery time when our framework is deployed on a measuring device and run the communication over radio networks.

REFERENCES

[1] G. Dan, K.-S. Lui, R. Tabassum, Q. Zhu, and K. Nahrstedt, "SELINDA: A Secure, Scalable and Light-Weight Data Collection Protocol for Smart Grids," to appear in IEEE SmartGridComm 2013.

[2] S. Sridhar, A. Hahn and M. Govindarasu, "Cyber–Physical System Security for the Electric Power Grid," in *Proceedings of the IEEE*, vol. 100, no.1, pp. 210-224, 2012.

[3] G. E. Suh and S. Devadas,"Physical unclonable functions for device authentication and secret key generation," In *Proceedings of Design Automation Conference*, pp. 9–14, 2007.

[4] M. Steiner, G. Tsudik and M. Waidner, "Refinement and extension of encrypted key exchange," *ACM SIGOPS*, vol. 29, no. 3, July 1995.

[5] T. Wu, "The secure remote password protocol," *Internet Society symposium on Network and Distributed System Security*, 1998.

[6] S. M. Bellovin and M. Merritt, "Augmented Encrypted Key Exchange: a Password Based Protocol Secure Against Dictionary Attacks and Password File Compromise," AT&T Bell Laboratories, 1994.

[7] L. von Ahn, M. Blum and J. Langford, "Telling Humans and Computers Apart (Automatically) or How Lazy Cryptographers do AI," *Communications of the ACM*, vol. 47, no. 2, pp. 57-60, 2004.

[8] D. P. Jablon, "Strong Password-Only Authenticated Key Exchange," *ACM Computer Communications Review*, October 1996.

[9] M. M. Fouda, Z. M. Fadlullah, N. Kato, R. Lu and X. Shen, "A Lightweight Message Authentication Scheme for Smart Grid Communications," in *Proceedings of IEEE Smart Grid*, vol.2, no.4, pp.675-685, 2011.

[10] Personal communication with Ameren, TCIPG industry Workshop, 2012, Urbana, IL.

[11] A. Shamir, "Identity-based Cryptosystems and Signature Schemes," in *Proceedings of CRYPTO*, Aug. 1985.

[12] H. K. H. So, S. H. M Kwok, E. Y. Lam, and K. Lui, "Zero-Configuration Identity-Based Signcryption Scheme for Smart Grid," in *Proceedings of IEEE International Conference on Smart Grid Communications*, pp.321-326, Oct. 2010.

[13] B. Y. Mo, T. H. Kim, K. Brancik, D. Dickinson, H. Lee, A. Perrig, and B. Sinopoli, "Cyber-physical security of a smart grid infrastructure," in *Proceedings of the IEEE*, vol. 100, no. 1, pp. 195–209, Jan. 2012.

[14] M. Nabeel, S. Kerr, X. Ding, and E. Bertino, "Authentication and Key Management for Advanced Metering Infrastructures Utilizing Physically Unclonable Functions," In *Proceedings of IEEE on Smart Grid Communications*, 2012.

[15] U. Rührmair, F. Sehnke, J. Sölter, G. Dror, S. Devadas, and J. Schmidhuber, "Modeling Attacks on Physical Unclonable Functions," in *Proceedings of the 17th ACM conference on Computer and communications security*, pp. 237-249, 2010.

[16] J. Zhou, Q. Hu, and Y. Qian, "Scalable distributed communication architectures to support advanced metering infrastructure in smart grid," IEEE Transactions on Parallel and Distributed Systems, vol. 23, no. 9, September 2012.

[17] N. Saputro and K. Akkaya, "Performance evaluation of smart grid data aggregation via homomorphic encryption," in Proc. of IEEE WCNC, 2012.

[18] D. Wu and C. Zhou, "Fault-tolerant and scalable key management for smart grid," IEEE Transactions on Smart Grid, vol. 2, no. 2, June 2011.

[19] T. Khalifa, K. Naik, M. Alsabaan, A. Nayak, and N. Goel, "Transport protocol for smart grid infrastructure," in Proc. of IEEE International Conference on Ubiquitous and Future Networks, 2010.

[20] Y.-J. Kim, V. Kolesnikov, H. Kim, and M. Thottan, "SSTP: a scalable and secure transport protocol for smart grid data collection," in *Proceedings of IEEE SmartGridComm*, 2011.

[21] B. Vaidya, D. Makrakis, and H. T. Mouftah, "Authentication and autho- rization mechanisms for substation automation in smart grid network," IEEE Network, pp. 5–11, Jan/Feb 2013.

[22] J. Liu, Y. Xiao, S. Li, W. Liang, and C. Chen, "Cyber security and privacy issues in smart grids," *IEEE Communications Surveys and Tutorials*, vol. 14, no. 4, Fourth Quarter 2012.

[23] G. Da´n, H. Sandberg, G. Bjrkman, and M. Ekstedt, "Challenges in power system information security," *IEEE Security & Privacy Magazine*, vol. 10, no. 4, 2012.

[24] M. M. Fouda, Z. M. Fadlullah, N. Kato, R. Lu, and X. Shen, "A lightweight message authentication scheme for smart grid communi- cations," *IEEE Transactions on Smart Grid*, vol. 2, no. 4, December 2011.

[25] C. Bekara, T. Luckenbach, and K. Bekara, "A privacy preserving and secure authentication protocol for the advanced metering infrastructure with non-repudiation service," in *Proceedings of ENERGY*, 2012.

[26] Y. Law, G. Kounga, and A. Lo, "WAKE: Key management scheme for wide-area measurement systems in smart grid*," IEEE Communications Magazine*, January 2013.

[27] N. Liu, J. Chen, L. Zhu, J. Zhan, and Y. He, “A key management scheme for secure communications of advanced metering infrastructure in smart grid,” *IEEE Transactions on Industrial Electronics*, to appear.

[28] IEEE 1815-2012, “Dnp3 secure authentication version 5,” 2011.

[29] RFC 5246, “The transport layer security (tls) protocol version 1.2,” 2008.

[30] C. Lim and P. Lee, “A key recovery attack on discrete log-based schemes using a prime order subgroup,” in *Proceedings of CRYPTO*, 1998.

[31] R. L. Rivest, A. Shamir, and L. Adleman, “A Method for Obtaining Digital Signatures and Public-Key Cryptosystems,” *Communications of the ACM*, vol. 21, pp. 120-126, Feb 1978.

[32] W. Diffie and M. E. Hellman, “Privacy and Authentication: An Introduction to cryptography,” in *Proceedings of the IEEE*, vol. 67, No. 3, pp. 397-427, March 1979.

[33] S. M. Bellovin and M. Merritt, “Encrypted Key Exchange: Password-Based Protocols Secure Against Dictionary Attacks,” in *Proceedings of the IEEE Symposium on Research in Security and Privacy*, 1992.

[34] "Announcing the ADVANCED ENCRYPTION STANDARD (AES)," *Federal Information Processing Standards Publication 197*, United States National Institute of Standards and Technology (NIST), November 26, 2001.

[35] “Cryptographic Hash,” http://csrc.nist.gov/groups/ST/hash/.

[36] “Enhancing Situational Awareness,” http://nyssmartgrid.com/innovation-highlights/ enhancing-situational-awareness/, accessed: 03/10/2013.

[37] R. Tabassum, K. Nahrstedt, E. Rogers and K. Lui, "SCAPACH: Scalable Password-Changing Protocol for Smart Grid Device Authentication," *Computer Communications and Networks (ICCCN), 2013 22nd International Conference*, pp.1-5, July 2013.

[38] I. Rouf, H. Mustafa, M. Xu, W. Wenyuan, R. Miller and M. Gruteser, “Neighborhood Watch: Security and Privacy Analysis of Automatic Meter Reading Systems,” in *proceedings of the 2012 ACM Conference on Computer and Communications Security*, pp. 462-473, 2012.

[39] A. Giani, E. Bitar, M. Garcia, M. McQueen, P. Khargonekar and K. Poolla, "Smart grid data integrity attacks: characterizations and countermeasures$^{\pi}$," *2011 IEEE International Conference on Smart Grid Communications (SmartGridComm)*, pp.232-237, 17-20 Oct 2011.

[40] T. Baumeister, “Adapting PKI for the smart grid,” *2011 IEEE International Conference on Smart Grid Communications (SmartGridComm)*, pp.249-254, 17-20 Oct. 2011.

[41] W. Burr, D Dodson, E. Newton, R. Perlner, W. Polk, S. Gupta and E. Nabbus, “Electronic Authentication Guideline,” *NIST SP - 800-63-2*, August 29, 2013.

[42] H. Khurana, M. Hadley, Ning Lu, D. Frincke, “Smart-grid security issues,” *Security & Privacy, IEEE*, vol.8, no.1, pp.81-85, Jan-Feb 2010.